

Matthew Mendoza - Assignment 04

CSC 174 - Section 02

- [Matthew Mendoza - Assignment 04](#)
 - [Section 4 - Function : Course_Instructor](#)
 - [Function Course_Instructor Call](#)
 - [Function Course_Instructor Result](#)
 - [Section 5 - Procedures](#)
 - [Procedure : Get_TA](#)
 - [Procedure : Get_TA Call](#)
 - [Procedure : Get_TA Result](#)
 - [Procedure : GetStudentCourse](#)
 - [Procedure : GetStudentCourse Call](#)
 - [Procedure : GetStudentCourse Result](#)

A PDF file include screenshots (in the following order):

1. Call the function and the results displayed.
2. Call the procedures and the results displayed.

Section 4 - Function : [Course_Instructor](#)

```

/* Function Course_Instructor
 * Input parameter : course name
 * Return : Instructor name
 */
DELIMITER $$
CREATE FUNCTION Course_Instructor (course_name VARCHAR(30))
RETURNS VARCHAR(20)
BEGIN
    DECLARE instructor_name VARCHAR(20);
    SELECT InstructorName INTO instructor_name
    FROM Instructor
    -- so we can use the CourseName input parameter in the
    -- WHERE clause of the outer query to return the InstructorName from the
    Instructor table
    WHERE InstructorID = (
        SELECT InstructorID
        FROM Course
        WHERE CourseName = course_name
    );
    RETURN instructor_name;
END $$
DELIMITER ;

```

Function [Course_Instructor](#) Call

```
SELECT Course_Instructor('Marketing Analytics'); -- Expects : "Byleth
Eisner"
SELECT Course_Instructor('Intro to Prob Theory'); -- Expects : "Byleth
Eisner"
SELECT Course_Instructor('Computer Forensics'); -- Expects : "Byleth
Eisner"
SELECT Course_Instructor('Combinatorics'); -- Expects : "Hanneman von
Essar"
SELECT Course_Instructor('Critical Thinking'); -- Expects : "Hanneman von
Essar"
SELECT Course_Instructor('Leadership and Management'); -- Expects : "Hanneman von
Essar"
SELECT Course_Instructor('Nursing Care Of Adults'); -- Expects : "Manuela
Casagrande"
SELECT Course_Instructor('Microbiology'); -- Expects : "Manuela
Casagrande"
SELECT Course_Instructor('Argumentation'); -- Expects : "Manuela
Casagrande"
SELECT Course_Instructor('Operations Management'); -- Expects : "Shamir
Nevrand"
SELECT Course_Instructor('Principles Of Marketing'); -- Expects : "Shamir
Nevrand"
SELECT Course_Instructor('Investments'); -- Expects : "Shamir
Nevrand"
SELECT Course_Instructor('Sustainability Business'); -- Expects : "Cassandra
Charon"
SELECT Course_Instructor('Venture Growth Strategies'); -- Expects : "Cassandra
Charon"
SELECT Course_Instructor('Corporate Entrepreneurship'); -- Expects : "Cassandra
Charon"
```

Function `Course_Instructor` Result

```
mysql> SELECT Course_Instructor('Marketing Analytics');      -- Expects : "Byleth Eisner"
+-----+
| Course_Instructor('Marketing Analytics') |
+-----+
| Byleth Eisner                            |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Intro to Prob Theory');    -- Expects : "Byleth Eisner"
+-----+
| Course_Instructor('Intro to Prob Theory') |
+-----+
| Byleth Eisner                            |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Computer Forensics');      -- Expects : "Byleth Eisner"
+-----+
| Course_Instructor('Computer Forensics') |
+-----+
| Byleth Eisner                            |
+-----+
1 row in set (0.00 sec)
```

```
mysql> SELECT Course_Instructor('Combinatorics');           -- Expects : "Hanneman von Essar"
+-----+
| Course_Instructor('Combinatorics') |
+-----+
| Hanneman von Essar                  |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Critical Thinking');        -- Expects : "Hanneman von Essar"
+-----+
| Course_Instructor('Critical Thinking') |
+-----+
| Hanneman von Essar                  |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Leadership and Management'); -- Expects : "Hanneman von Essar"
+-----+
| Course_Instructor('Leadership and Management') |
+-----+
| Hanneman von Essar                  |
+-----+
1 row in set (0.00 sec)
```

```
mysql> SELECT Course_Instructor('Nursing Care Of Adults');      -- Expects : "Manuela Casagranda"
+-----+
| Course_Instructor('Nursing Care Of Adults') |
+-----+
| Manuela Casagranda                          |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Microbiology');               -- Expects : "Manuela Casagranda"
+-----+
| Course_Instructor('Microbiology') |
+-----+
| Manuela Casagranda                  |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Argumentation');              -- Expects : "Manuela Casagranda"
+-----+
| Course_Instructor('Argumentation') |
+-----+
| Manuela Casagranda                  |
+-----+
1 row in set (0.00 sec)
```

```
mysql> SELECT Course_Instructor('Operations Management');      -- Expects : "Shamir Nevrand"
+-----+
| Course_Instructor('Operations Management') |
+-----+
| Shamir Nevrand                            |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Principles Of Marketing');    -- Expects : "Shamir Nevrand"
+-----+
| Course_Instructor('Principles Of Marketing') |
+-----+
| Shamir Nevrand                              |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Investments');                -- Expects : "Shamir Nevrand"
+-----+
| Course_Instructor('Investments') |
+-----+
| Shamir Nevrand                      |
+-----+
1 row in set (0.00 sec)
```

```
mysql> SELECT Course_Instructor('Sustainability Business'); -- Expects : "Cassandra Charon"
+-----+
| Course_Instructor('Sustainability Business') |
+-----+
| Cassandra Charon |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Venture Growth Strategies'); -- Expects : "Cassandra Charon"
+-----+
| Course_Instructor('Venture Growth Strategies') |
+-----+
| Cassandra Charon |
+-----+
1 row in set (0.00 sec)

mysql> SELECT Course_Instructor('Corporate Entrepreneurship'); -- Expects : "Cassandra Charon"
+-----+
| Course_Instructor('Corporate Entrepreneurship') |
+-----+
| Cassandra Charon |
+-----+
1 row in set (0.00 sec)
```

Section 5 - Procedures

Procedure : **Get_TA**

```
/* Logic for procedure Get_TA
Take Instructor_ID as input parameter
-- Begin procedure body
-- Print TA names
-- Course, Student, TA tables are joined together to get TA names for a given
instructor
-- InstructorID is a foreign key in Course table and a primary key in
Instructor table
-- CAST() function converts VARCHAR to INT for comparison with INT column in
Course table
-- End of procedure body
*/
DELIMITER $$ -- Change delimiter to $$ to allow ; in procedure body
CREATE PROCEDURE Get_TA(IN Instructor_ID VARCHAR(20))
BEGIN
    SELECT S.StudentName
    FROM Course AS C, Student AS S, TA AS T
    WHERE C.InstructorID=CAST(Instructor_ID AS UNSIGNED INTEGER) AND C.TASSN=T.SSN
    AND T.SSN=S.SSN;
END $$ -- End of procedure body
DELIMITER ; -- Change delimiter back to ;
```

Procedure : **Get_TA** Call

```
CALL Get_TA("0900000001"); -- Expected : Claude and Edelgard
CALL Get_TA("0900000002"); -- Expected : Dimitri and Edelgard
CALL Get_TA("0900000003"); -- Expected : Claude and Dimitri
CALL Get_TA("0900000004"); -- Expected : Claude and Dimitri and Edelgard
CALL Get_TA("0900000005"); -- Expected : Claude and Dimitri and Edelgard
```

Procedure : Get_TA Result

```
mysql> CALL Get_TA("0900000001"); -- Expected : Claude and Edelgard
+-----+
| StudentName |
+-----+
| Claude von Riegan |
| Claude von Riegan |
| Edelgard von Hresvelg |
+-----+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL Get_TA("0900000002"); -- Expected : Dimitri and Edelgard
+-----+
| StudentName |
+-----+
| Dimitri Alexandre Blaiddyd |
| Edelgard von Hresvelg |
| Edelgard von Hresvelg |
+-----+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL Get_TA("0900000003"); -- Expected : Claude and Dimitri
+-----+
| StudentName |
+-----+
| Claude von Riegan |
| Dimitri Alexandre Blaiddyd |
| Dimitri Alexandre Blaiddyd |
+-----+
3 rows in set (0.00 sec)
```

```
mysql> CALL Get_TA("0900000004"); -- Expected : Claude and Dimitri and Edelgard
+-----+
| StudentName |
+-----+
| Claude von Riegan |
| Dimitri Alexandre Blaiddyd |
| Edelgard von Hresvelg |
+-----+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL Get_TA("0900000005"); -- Expected : Claude and Dimitri and Edelgard
+-----+
| StudentName |
+-----+
| Claude von Riegan |
| Dimitri Alexandre Blaiddyd |
| Edelgard von Hresvelg |
+-----+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Procedure : **GetStudentCourse**

```
/* Logic for procedure GetStudentCourse
Take Course_Number as input parameter
-- Begin procedure body
-- Print SSN, Student Name, Email
-- Course, Student, Enrolled tables are joined together to get student info for
a given course
-- Course_Number is a foreign key in Enrolled table and a primary key in Course
table
-- End of procedure body
*/
DELIMITER $$
CREATE PROCEDURE GetStudentCourse(IN Course_Number INTEGER)
BEGIN
SELECT S.SSN, S.StudentName, S.Address, S.Email
FROM Course AS C, Enrolled AS E, Student AS S
WHERE Course_Number=C.CourseNo AND C.CourseNo=E.CourseNo AND E.SSN=S.SSN ;
END $$
DELIMITER ;
```

Procedure : **GetStudentCourse** Call

```
CALL GetStudentCourse('3300001'); -- Marketing Analytics -> Expect 8 students
CALL GetStudentCourse('3300002'); -- Intro to Prob Theory -> Expect 9 students
CALL GetStudentCourse('3300003'); -- Computer Forensics -> Expect 7 students
CALL GetStudentCourse('3300004'); -- Combinatorics -> Expect 9 students
CALL GetStudentCourse('3300005'); -- Critical Thinking -> Expect 8 students
CALL GetStudentCourse('3300006'); -- Leadership and Management -> Expect 11
students
CALL GetStudentCourse('3300007'); -- Nursing Care Of Adults -> Expect 9 students
CALL GetStudentCourse('3300008'); -- Microbiology -> Expect 7 students
CALL GetStudentCourse('3300009'); -- Argumentation -> Expect 10 students
CALL GetStudentCourse('3300010'); -- Operations Management -> Expect 11 students
CALL GetStudentCourse('3300011'); -- Principles Of Marketing -> Expect 7 students
CALL GetStudentCourse('3300012'); -- Investments -> Expect 9 students
CALL GetStudentCourse('3300013'); -- Sustainability Business -> Expect 1 Student
CALL GetStudentCourse('3300014'); -- Venture Growth Strategies -> Expect 1 Student
```

```
CALL GetStudentCourse('3300015'); -- Corporate Entrepreneurship -> Expect 1
Student
```

Procedure : GetStudentCourse Result

```
mysql> CALL GetStudentCourse('3300001'); -- Marketing Analytics -> Expect 8 students
```

SSN	StudentName	Address	Email
190000001	Haruhi Fujioka	7098 Prince Ave	haruhi_fujioka@hostclub.com
190000006	Ferdinand von Aegir	123 House Aegir	ferdinand_von_aegir@aegir.com
190000008	Dorothea Arnault	941 Mittelfrank Opera Company	dorothea@mittelfrank_opera.com
190000010	Mercedes von Bartels	321 Garreg Mach Monastery	mercie@garregmach.com
190000011	Annette Fantine Dominic	572 House Dominic	annette@dominic.com
190000014	Raphael Kirsten	8544 Leicester	the_beast@leicester.com
190000015	Yuri Leclerc	8982 Rowe Ave	Yuri@ashen_wolves.com
200100003	Edelgard von Hresvelg	8756 Enbarr	flame_emperor@adrestian.com

```
8 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL GetStudentCourse('3300002'); -- Intro to Prob Theory -> Expect 9 students
```

SSN	StudentName	Address	Email
190000001	Haruhi Fujioka	7098 Prince Ave	haruhi_fujioka@hostclub.com
190000003	Kyoya Ootori	756 Pennington Road	kyoya_ootori@hostclub.com
190000005	Takashi Morinozuka	9245 Pineknoll Ave	takashi_nozuka@hostclub.com
190000006	Ferdinand von Aegir	123 House Aegir	ferdinand_von_aegir@aegir.com
190000008	Dorothea Arnault	941 Mittelfrank Opera Company	dorothea@mittelfrank_opera.com
190000009	Felix Hugo Fraldarius	123 House Fraldarius	felix@fraldarius.com
190000015	Yuri Leclerc	8982 Rowe Ave	Yuri@ashen_wolves.com
200100001	Claude von Riegan	4592 Leicester Federation	claud@leicester.com
200100002	Dimitri Alexandre Blaiddyd	1462 Garreg Mach Monastery	the_tempest_king@fhirdiad.com

```
9 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL GetStudentCourse('3300003'); -- Computer Forensics -> Expect 7 students
```

SSN	StudentName	Address	Email
190000001	Haruhi Fujioka	7098 Prince Ave	haruhi_fujioka@hostclub.com
190000002	Tamaki Suoh	743 Arlington Lane	tamaki_suoh@hostclub.com
190000003	Kyoya Ootori	756 Pennington Road	kyoya_ootori@hostclub.com
190000008	Dorothea Arnault	941 Mittelfrank Opera Company	dorothea@mittelfrank_opera.com
190000010	Mercedes von Bartels	321 Garreg Mach Monastery	mercie@garregmach.com
190000013	Marianne von Edmund	8254 House Edmund	Marianne@Edmund.com
200100002	Dimitri Alexandre Blaiddyd	1462 Garreg Mach Monastery	the_tempest_king@fhirdiad.com

```
7 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```



```
mysql> CALL GetStudentCourse('3300004'); -- Combinatorics -> Expect 9 students
```

SSN	StudentName	Address	Email
190000002	Tamaki Suoh	743 Arlington Lane	tamaki_suoh@hostclub.com
190000003	Kyoya Ootori	756 Pennington Road	kyoya_ootori@hostclub.com
190000005	Takashi Morinozuka	9245 Pineknoll Ave	takashi_nozuka@hostclub.com
190000006	Ferdinand von Aegir	123 House Aegir	ferdinand_von_aegir@aegir.com
190000007	Bernadetta von Varley	123 House Varley	bernie@varley.com
190000009	Felix Hugo Fraldarius	123 House Fraldarius	felix@fraldarius.com
190000011	Annette Fantine Dominic	572 House Dominic	annette@dominic.com
190000013	Marianne von Edmund	8254 House Edmund	Marianne@Edmund.com
200100001	Claude von Riegan	4592 Leicester Federation	claudio@leicester.com

```
9 rows in set (0.00 sec)
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL GetStudentCourse('3300005'); -- Critical Thinking -> Expect 8 students
```

SSN	StudentName	Address	Email
190000002	Tamaki Suoh	743 Arlington Lane	tamaki_suoh@hostclub.com
190000003	Kyoya Ootori	756 Pennington Road	kyoya_ootori@hostclub.com
190000006	Ferdinand von Aegir	123 House Aegir	ferdinand_von_aegir@aegir.com
190000010	Mercedes von Bartels	321 Garreg Mach Monastery	mercier@garregmach.com
190000012	Lysithea von Ordellia	9245 House Ordellia	lys@ordellia.com
190000013	Marianne von Edmund	8254 House Edmund	Marianne@Edmund.com
200100001	Claude von Riegan	4592 Leicester Federation	claudio@leicester.com
200100003	Edelgard von Hresvelg	8756 Enbarr	flame_emperor@adrestian.com

```
8 rows in set (0.00 sec)
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> CALL GetStudentCourse('3300006'); -- Leadership and Management -> Expect 11 students
```

SSN	StudentName	Address	Email
190000003	Kyoya Ootori	756 Pennington Road	kyoya_ootori@hostclub.com
190000004	Mitsukuni Haninozuka	142 Old Vernon St	mitsukuni_nozuka@hostclub.com
190000005	Takashi Morinozuka	9245 Pineknoll Ave	takashi_nozuka@hostclub.com
190000007	Bernadetta von Varley	123 House Varley	bernie@varley.com
190000009	Felix Hugo Fraldarius	123 House Fraldarius	felix@fraldarius.com
190000011	Annette Fantine Dominic	572 House Dominic	annette@dominic.com
190000012	Lysithea von Ordellia	9245 House Ordellia	lys@ordellia.com
190000013	Marianne von Edmund	8254 House Edmund	Marianne@Edmund.com
200100001	Claude von Riegan	4592 Leicester Federation	claudio@leicester.com
200100002	Dimitri Alexandre Blaiddyd	1462 Garreg Mach Monastery	the_tempest_king@fhirdiad.com
200100003	Edelgard von Hresvelg	8756 Enbarr	flame_emperor@adrestian.com

```
11 rows in set (0.00 sec)
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL GetStudentCourse('3300007'); -- Nursing Care Of Adults -> Expect 9 students
```

SSN	StudentName	Address	Email
190000001	Haruhi Fujioka	7098 Prince Ave	haruhi_fujioka@hostclub.com
190000004	Mitsukuni Haninozuka	142 Old Vernon St	mitsukuni_nozuka@hostclub.com
190000006	Ferdinand von Aegir	123 House Aegir	ferdinand_von_aegir@aegir.com
190000007	Bernadetta von Varley	123 House Varley	bernie@varley.com
190000008	Dorothea Arnault	941 Mittelfrank Opera Company	dorothea@mittelfrank_opera.com
190000010	Mercedes von Bartels	321 Garreg Mach Monastery	mercie@garregmach.com
190000014	Raphael Kirsten	8544 Leicester	the_beast@leicester.com
190000015	Yuri Leclerc	8982 Rowe Ave	Yuri@ashen_wolves.com
200100001	Claude von Riegan	4592 Leicester Federation	claudio@leicester.com

```
9 rows in set (0.00 sec)
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL GetStudentCourse('3300008'); -- Microbiology -> Expect 7 students
```

SSN	StudentName	Address	Email
190000001	Haruhi Fujioka	7098 Prince Ave	haruhi_fujioka@hostclub.com
190000004	Mitsukuni Haninozuka	142 Old Vernon St	mitsukuni_nozuka@hostclub.com
190000005	Takashi Morinozuka	9245 Pineknoll Ave	takashi_nozuka@hostclub.com
190000008	Dorothea Arnault	941 Mittelfrank Opera Company	dorothea@mittelfrank_opera.com
190000009	Felix Hugo Fraldarius	123 House Fraldarius	felix@fraldarius.com
190000014	Raphael Kirsten	8544 Leicester	the_beast@leicester.com
200100002	Dimitri Alexandre Blaiddyd	1462 Garreg Mach Monastery	the_tempest_king@fhirdiad.com

```
7 rows in set (0.00 sec)
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL GetStudentCourse('3300009'); -- Argumentation -> Expect 10 students
```

SSN	StudentName	Address	Email
190000002	Tamaki Suoh	743 Arlington Lane	tamaki_suoh@hostclub.com
190000004	Mitsukuni Haninozuka	142 Old Vernon St	mitsukuni_nozuka@hostclub.com
190000007	Bernadetta von Varley	123 House Varley	bernie@varley.com
190000010	Mercedes von Bartels	321 Garreg Mach Monastery	mercie@garregmach.com
190000011	Annette Fantine Dominic	572 House Dominic	annette@dominic.com
190000012	Lysithea von Ordelia	9245 House Ordelia	lys@ordelia.com
190000013	Marianne von Edmund	8254 House Edmund	Marianne@Edmund.com
190000014	Raphael Kirsten	8544 Leicester	the_beast@leicester.com
200100002	Dimitri Alexandre Blaiddyd	1462 Garreg Mach Monastery	the_tempest_king@fhirdiad.com
200100003	Edelgard von Hresvelg	8756 Enbarr	flame_emperor@adrestian.com

```
10 rows in set (0.00 sec)
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL GetStudentCourse('3300010'); -- Operations Management -> Expect 11 students
```

SSN	StudentName	Address	Email
190000001	Haruhi Fujioka	7098 Prince Ave	haruhi_fujioka@hostclub.com
190000002	Tamaki Suoh	743 Arlington Lane	tamaki_suoh@hostclub.com
190000004	Mitsukuni Haninozuka	142 Old Vernon St	mitsukuni_nozuka@hostclub.com
190000005	Takashi Morinozuka	9245 Pineknoll Ave	takashi_nozuka@hostclub.com
190000007	Bernadetta von Varley	123 House Varley	bernie@varley.com
190000009	Felix Hugo Fraldarius	123 House Fraldarius	felix@fraldarius.com
190000011	Annette Fantine Dominic	572 House Dominic	annette@dominic.com
190000012	Lysithea von Ordelia	9245 House Ordelia	lys@ordelia.com
190000014	Raphael Kirsten	8544 Leicester	the_beast@leicester.com
190000015	Yuri Leclerc	8982 Rowe Ave	Yuri@ashen_wolves.com
200100003	Edelgard von Hresvelg	8756 Enbarr	flame_emperor@adrestian.com

11 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

```
mysql> CALL GetStudentCourse('3300011'); -- Principles Of Marketing -> Expect 7 students
```

SSN	StudentName	Address	Email
190000002	Tamaki Suoh	743 Arlington Lane	tamaki_suoh@hostclub.com
190000006	Ferdinand von Aegir	123 House Aegir	ferdinand_von_aegir@aegir.com
190000007	Bernadetta von Varley	123 House Varley	bernie@varley.com
190000010	Mercedes von Bartels	321 Garreg Mach Monastery	mercie@garregmach.com
190000011	Annette Fantine Dominic	572 House Dominic	annette@dominic.com
190000012	Lysithea von Ordelia	9245 House Ordelia	lys@ordelia.com
190000013	Marianne von Edmund	8254 House Edmund	Marianne@Edmund.com
190000015	Yuri Leclerc	8982 Rowe Ave	Yuri@ashen_wolves.com

8 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

```
mysql> CALL GetStudentCourse('3300012'); -- Investments -> Expect 9 students
```

SSN	StudentName	Address	Email
190000003	Kyoya Ootori	756 Pennington Road	kyoya_ootori@hostclub.com
190000004	Mitsukuni Haninozuka	142 Old Vernon St	mitsukuni_nozuka@hostclub.com
190000005	Takashi Morinozuka	9245 Pineknoll Ave	takashi_nozuka@hostclub.com
190000008	Dorothea Arnault	941 Mittelfrank Opera Company	dorothea@mittelfrank_opera.com
190000009	Felix Hugo Fraldarius	123 House Fraldarius	felix@fraldarius.com
190000012	Lysithea von Ordelia	9245 House Ordelia	lys@ordelia.com
190000014	Raphael Kirsten	8544 Leicester	the_beast@leicester.com
190000015	Yuri Leclerc	8982 Rowe Ave	Yuri@ashen_wolves.com

8 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

```
mysql> CALL GetStudentCourse('3300013'); -- Sustainability Business -> Expect 1 Student
+-----+-----+-----+-----+
| SSN      | StudentName          | Address                | Email                                |
+-----+-----+-----+-----+
| 200100002 | Dimitri Alexandre Blaiddyd | 1462 Garreg Mach Monastery | the_tempest_king@fhirdiad.com |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL GetStudentCourse('3300014'); -- Venture Growth Strategies -> Expect 1 Student
+-----+-----+-----+-----+
| SSN      | StudentName          | Address                | Email                                |
+-----+-----+-----+-----+
| 200100003 | Edelgard von Hresvelg | 8756 Enbarr           | flame_emperor@adrestian.com |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL GetStudentCourse('3300015'); -- Corporate Entrepreneurship -> Expect 1 Student
+-----+-----+-----+-----+
| SSN      | StudentName          | Address                | Email                                |
+-----+-----+-----+-----+
| 200100001 | Claude von Riegan    | 4592 Leicester Federation | clauder@leicester.com |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```